Design of Large-Scale Metaheuristic Component Studies

Helena Stegherr
University of Augsburg, Organic Computing Group
Augsburg, Germany
helena.stegherr@informatik.uni-augsburg.de

Michael Heider
University of Augsburg, Organic Computing Group
Augsburg, Germany
michael.heider@informatik.uni-augsburg.de

Leopold Luley
University of Augsburg, Organic Computing Group
Augsburg, Germany
leopold.luley@student.uni-augsburg.de

Jörg Hähner
University of Augsburg, Organic Computing Group
Augsburg, Germany
joerg.haehner@informatik.uni-augsburg.de

ABSTRACT

Metaheuristics employ a variety of different components using a wide array of operators to execute their search. This determines their intensification, diversification and all other behavioural features and is thus critical for success on different optimisation problems. Choosing the right components with the right operators remains a difficult task. In this paper we propose a design of experiments that should be used for extensive component studies. We demonstrate the applicability of this design by exploring the differences in operator specific performance in two closely related metaheuristic frameworks—the well-known $(\mu + \lambda)$-Evolution Strategy and the strongly metaphor-focussed Invasive Weed Optimisation—where operators show varying degrees of similarity in different components. This experiment shows that similarity of operators does not comprehensively account for similarity in performance. Presumably small changes of an operator can influence the algorithmic behaviour more than the utilisation of a completely different operator in another component. Even when employed in different combinations, these influences remain strong. This emphasises the need for a more detailed analysis of the specific effects of components and their respective operators on the search process.

CCS CONCEPTS

• **Computing methodologies** → *Heuristic function construction*;

KEYWORDS

Metaheuristics, Component analysis, Design of experiments

ACM Reference Format:
Helena Stegherr, Michael Heider, Leopold Luley, and Jörg Hähner. 2021. Design of Large-Scale Metaheuristic Component Studies. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3449726.3463168

1 INTRODUCTION

The performance of a metaheuristic varies dependent on the given optimisation problem. This variation not only occurs respective to the metaheuristic framework but also to the components included in this framework, i. e. the utilised operators and the conditions for their usage. Furthermore, many components and their operators are not exclusive to one single metaheuristic but are incorporated in a range of different frameworks.

This creates similarities between metaheuristics with very different descriptions and inspirational aspects and results in the question if it is possible to describe metaheuristics in a unified way [3, 18]. A unified metaheuristic framework can then provide the basis to construct algorithms from the set of components, customised for the problem at hand. This strongly relates to approaches of metaheuristic hybridisation and hyperheuristics, which could both profit from prior knowledge of the influences of components. However, this requires extensive analysis of the influences of metaheuristic components, in terms of their concepts, their behaviour and the resulting performance [18]. In addition, the combination of certain components can cause a different behaviour as their individual evaluations would suggest [44], thus extending the required analysis.

There are some studies on metaheuristic components concerning the incorporation of different operators into the framework and how their combinations influence the overall performance (e. g. [38, 44]). Furthermore, a number of studies has been performed analysing the hyperparameters of a metaheuristic, which are often part of an operator or provide conditions for its usage (e. g. [26]). However, these existing studies are insufficient as they have some major restrictions:

(1) Most studies analyse components of and in the same metaheuristic frameworks, e. g. the *Genetic Algorithm*, *Evolution Strategy* and, more rarely, *Particle Swarm Optimisation.*
(2) Components and operators are often only evaluated on few optimisation problems and their instances.
(3) Not all components have been included in previous evaluations. This is especially a problem in light of the vast number of different generally applicable and problem-dependent operators.
(4) Combinations of different operators have not been analysed in detail, although the few available studies show that they can have strong influences on the performance.

(5) The influences of components are mostly evaluated with respect to the overall performance. Their behaviour during the search process is often not analysed.

Gaining a better understanding of metaheuristic components by extending the existing analysis can positively affect the applicability of metaheuristics for a given problem. At that, it is important to include not only well-known approaches, but also those heavily relying on their underlying metaphor, to prevent overlooking new, efficient strategies. Operators can be incorporated utilising the information on their behaviour and the search for suitable hyperparameters may be facilitated through prior knowledge. Furthermore, new combinations of operators from different metaheuristics can result in effective hybrid approaches and novel metaheuristic frameworks can be built by expanding the range of component combinations.

To this end, an analysis of how metaheuristic components influence the search behaviour of the algorithms is required [53]. For a comprehensive understanding of these influences, the following questions need to be answered:

RQ-1 How do the components of metaheuristics relate to the search behaviour, e. g. exploration and exploitation, the solution quality, the convergence of the algorithm, etc.?
RQ-2 How do interactions/combinations of operators influence the search?
RQ-3 How can we quantify and evaluate these influences?
RQ-4 Is it possible to derive general statements on the behaviour and applicability of the components?

In this paper, we present a strategy for the execution of large-scale studies of metaheuristic components to provide insights related to these research questions. We want to focus on operators and their combinations, initially restricting the hyperparameter studies to the necessary minimum. Therefore, in Section 2 we describe existing approaches to the evaluation of component behaviour and influences. Section 3 presents descriptions and definitions for components and operators in the context of a conceptual framework for their interactions. Then we will illustrate how large-scale operator studies should be performed (Section 4). This includes the required assumptions (Section 4.1), the design of the experiments (Section 4.2) and a first evaluation to show the feasibility of our approach (Section 4.3). In the end, we summarise our findings (Section 5) and provide an overview on future work on the topic (Section 6).

2 RELATED WORK

Operators have been researched primarily for *Evolutionary Algorithms*. For these metaheuristics, there is a large set of different operators, resulting from extensions to the original algorithms and adaptations for specific problems.

The analysis of these operators can be divided into two areas: comparing the concepts of the operators, and performing studies to assess the performance of an algorithm utilising one or more specific operators. Reviews on operator descriptions exist on many different mutation and crossover variants [32, 40, 48, 49, 52], as well as on different selection operators [43, 46]. Studies on the performance influences an operator provides include the analysis of different mutation operators [8, 12, 41], crossover operators [1, 41] and combinations of both operators [38, 54], and often evaluate only the change in overall algorithmic performance on one or few optimisation problems. Analysis of the selection operators in *Genetic Algorithms* (GA) and *Evolution Strategies* (ES) is more detailed, providing further evaluations of the loss of diversity and selection intensity [10], the population diversity and the selection pressure [29] for the GA, and of the selection pressure in the ES [2], respectively. Only very few analyses include all three components in different variants [44].

There are some studies comparing components and certain operators in different metaheuristic frameworks [37]. However, framework-independent studies of operators are not yet in the focus of research. They require a notion of a unified framework that can be used to compose several different metaheuristics from a basic set of components, including operators. One approach for a generalised metaheuristic model and a resulting operator analysis is provided by Cruz-Duarte et al. [18]. They construct and evaluate 20 different metaheuristics from the same basic set of components. Furthermore, unified metaheuristic frameworks were used to improve *Differential Evolution* [39] and *Particle Swarm Optimisation* [20] by utilising operators from other metaheuristics. Other unification approaches are focused on a conceptual comparison [3], the detection of similarities between different metaheuristics [14, 19] or the construction of new algorithms from the same basic building blocks [47].

Another strategy to unify metaheuristic concepts is provided by Chicco and Mazza [16] and Batrinu et al. [6], focussing on finding more detailed common principles instead of a general algorithmic structure. These principles describe common capabilities of metaheuristic components that are required in a metaheuristic. They can be extended or specified by other categorisations of components, including their intensification and diversification capabilities [11] and the categorisation according to their search behaviour [33, 34]. Other criteria to categorise components according to their behaviour or their influence on the overall metaheuristic can be found in work on classification of metaheuristics (e.g. [36, 50, 51]). Furthermore, research on metaheuristic design patterns provides common structures for components [30].

3 METAHEURISTIC COMPONENTS

In this section, we clarify our understanding of metaheuristic components, operators and a unified or general metaheuristic framework. Furthermore, we describe how the components can be integrated into such a general framework so that it enables conducting operator studies. To this end, we use existing approaches as a basis and extend or adapt them if necessary.

Following Blum and Roli [11] and Lones [33, 34], we use the term *component* to describe any part of a metaheuristic that represents a recurring scheme and is related to intensification or diversification or any other behavioural feature of the metaheuristic. Components consist of *operators* that determine their behaviour, and parameters that determine the behaviour of the operators, respectively.

The different types of components can be derived from unification concepts. These often structure metaheuristics into distinguishable functional parts. However, the current approaches differ in denotation and level of detail. Bandaru and Deb [3] and de Armas et al. [19] use similar descriptions for a unified metaheuristic

framework. Central aspects are component structures for selection, variation and replacement and update of solutions. Bandaru and Deb [3] further add a structure for termination of the algorithm, while de Armas et al. [19] include the initialisation and the usage of an archive or memory. Furthermore, de Armas et al. [19] focus on the different sets of solutions at the stages between the structures. Cruz-Duarte et al. [18] use a different approach, where a combination of two structures, the perturbator and the selector, is repeated several times between initalisation and termination. Perturbator and selector can be used to describe any component ascribed to selection, variation and replacement of solutions. The approach by Song and Fong [47] is somewhere in between these strategies, consisting of initialisation, searching method, environmental response and adaptive method.

While the approaches of Cruz-Duarte et al. [18] and Song and Fong [47] are well suited as a template for implementation, they require additional information to conceptually analyse the utilised components. An approach differentiating the components in detail can facilitate both, the understanding of the resulting algorithmic concepts and the implementation and evaluation of different combinations. We will therefore combine the approaches of Bandaru and Deb [3] and de Armas et al. [19] to describe the possible types of components and to obtain a template for a structured analysis (see Figure 1).

The identified relevant structures of a general metaheuristic framework are the *initialisation*, *selection*, *generation*, *replacement*, *update*, *archiving* and *termination*. The naming of the structures is based on Bandaru and Deb [3] as these present more intuitive descriptions of the functions. *Initialisation* is not present as a component structure in Bandaru and Deb [3], while in de Armas et al. [19] it is termed generation method. It is included as the initialisation component influences the search process by its specification, e. g. if it is random or deterministic and how this is achieved. The *selection* structure (*output functions* in [19]) determines which solutions provide the basis for the *generation* (or *updating mechanism* in [19]) of new solutions. *Replacement* and *update* define which old solutions will be replaced and which new solutions are kept for the next iteration and can be combined to a single structure (*input functions* in [19]). The *archiving* structure acts as a memory for solutions and allows using them to generate new solutions. Finally, the *termination* determines when to stop the search process. All structures are composed of one or more components, except *archiving*, which is optional. However, there is often exactly one component for each structure present in a metaheuristic, although employing several components for *generation* is common.

The structures of a unified framework allow the categorisation of components according to their general functions in a metaheuristic. Furthermore, components can be categorised according to several sets of criteria. These include their intensifying and diversifying effects depending, for example, on how strongly the component is guided by the objective function, other functions or randomness [11]. Additionally, common underlying principles [6, 16] and recurring concepts [33, 34] can highlight influences and functional similarities of components. Chicco and Mazza [16] list the common principles as *Parallelism*, *Acceptance*, *Elitism*, *Selection*, *Decay/Reinforcement*, *Immunity*, *Self-adaptivity* and *Topology*. Recurring concepts are, according to Lones [33, 34], *Neighbourhood Search*, *Hill Climbing*, *Accepting Negative Moves*, *Restarts*, *Adaptive Memory Programming*, *Population-based Search*, *Intermediate Search*, *Directed Search*, *Variable Neighbourhood Search* and *Search Space Mapping*. An important aspect is that these principles or concepts are not necessarily attributable to individual components but can also result from combinations of components.

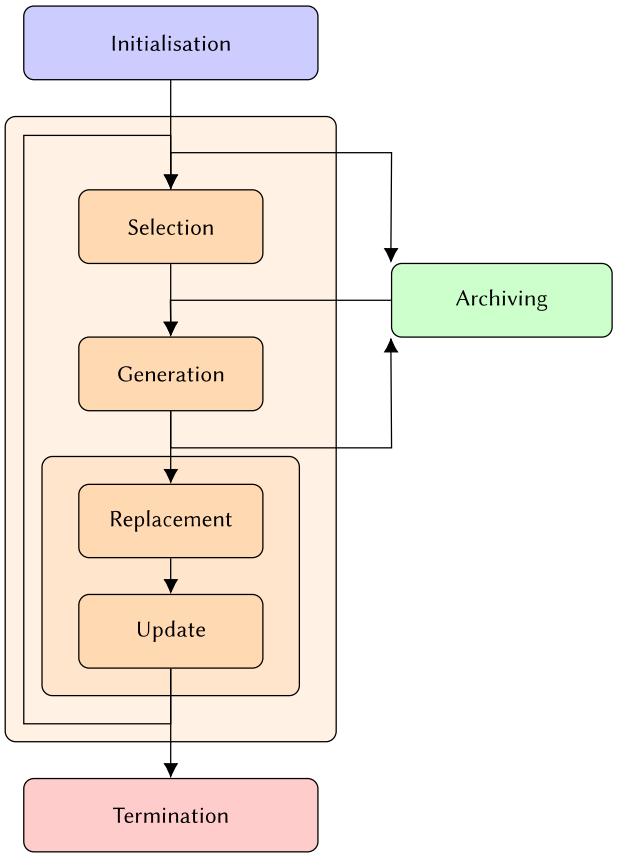

Figure 1: Component Structure of Metaheuristics (adapted from [3, 19]).

4 ANALYSIS OF METAHEURISTIC COMPONENTS

This section outlines an approach on the analysis of metaheuristic components. After stating the assumptions this analysis is based on, the experimental design is elucidated. Furthermore, an example for the execution of the analysis is presented.

4.1 Assumptions

We assume that all (or at least most) metaheuristics can be (re-)built by inserting components into a general framework and that this process does, if done right, neither influence the overall performance nor the behaviour of the metaheuristic. This is required for constructing a common basis for comparisons. Furthermore, we assume that we can utilise all components in all metaheuristics, as long as the essential structure of a general framework is preserved.

If this does not apply, it is not possible to compare components in different frameworks. Our last assumption is that the composition of components and the specific operators and parameters account for the search behaviour and the performance on a given problem. The refutation of this final assumption would necessitate further discussions about the applicability of this kind of study.

4.2 Design of Experiments

The experimental design of metaheuristic component studies is planned according to general guidelines for design of experiments [4, 9, 27, 28] and benchmarking [5, 31].

Goals: The overarching goal of the studies is to understand the influence of algorithmic components, specifically the individual operators, in terms of performance and search behaviour. This, however, is an abstract goal that needs to be specified by posing several questions directly connected to it. Among those are the research questions from section 1, but many more questions can be stated, for example:

- Are there operators that always result in at least "good" performance?
- Are there operators that always produce poor results?
- Which operators cause the same behaviour (at least in certain aspects)?
- Are problem-specific operators always better (on the specific problem they are intended for) than universally applicable ones?
- Is it possible to match operators to certain problem features?
- Do some operators perform better for a given problem than others, even with the best possible parameter settings?

Each of these questions requires specific measurements of performance and behaviour and suitable statistics for the evaluation. It is imperative to clearly state these for the respective analysis.

Problems: The next step is to find suitable optimisation problems for the analysis. These have to be diverse in terms of their difficulty to make performance differences of the components visible. Furthermore, they should possess different and preferably known characteristics or features that could require specific behaviour for good performance.

First evaluations should therefore be performed on standard benchmark problems, e. g. *Black-Box Optimization Benchmarking* (BBOB) functions [25], the *Travelling Salesperson Problem* (TSP) [42] and the *Quadratic Assignment Problem* (QAP) [13]. Later on, an extension to some real-world problems is necessary to see if the conclusions still hold. This is especially interesting when trying to match components to specific problem features or when determining generally well-functioning components.

Algorithms: Components cannot be evaluated without integration into an algorithm. To this end, all components will be integrated into the general metaheuristic framework described in section 3. This integration comprises two different approaches: the reconstruction of existing metaheuristics from the components and their specific operators and the construction of additional algorithms by new component combinations or operators. However, no comprehensive analysis is possible without considering the respective hyperparameters that guide the utilisation of operators. The necessary hyperparameter studies enable the establishing of a suitable parameter set to make operator combinations comparable. Furthermore, their influences on the performance as well as the behaviour of the algorithms can be examined.

Performance: The components have to be evaluated in terms of their influence on the performance but also the search behaviour, depending on the question to be answered or specific hypothesis stated. This requires different measures, with an additional distinction of those purely for continuous or discrete problems. For some established measures for discrete problems, it should be evaluated if they can be utilised for continuous problems as well.

A comprehensive list of measures with a focus on efficiency and effectiveness is provided by Halim et al. [24] while Scheibenpflug et al. [45] list suitable behavioural measures for combinatorial optimisation problems. Some measures can naturally be utilised for both, the assessment of performance and behaviour.

Appropriate measures for the performance relate to the solution quality, the computational budget, and the robustness of the algorithm [5]. These include, among others, the difference to the (known) optimum, the number of function evaluations, the number of successful runs, and statistics. Furthermore, the convergence measures are of interest [24], though they also relate to the behaviour of the algorithm. Behavioural measures include, the population diversity [15], selection pressure [23], exploration and exploitation steps [17], and amount of improvement, coverage of solution space and intensification and diversification ratios [45].

Analysis: The resulting measures for performance and behaviour must then be analysed to answer the research questions. In this case of component comparison, multiple algorithms are evaluated on multiple problems. Furthermore, for each research question and each measure, there has to be a clearly specified hypothesis. The first step is the exploratory data analysis, with the evaluation of mean, median, best, worst, first quartile, third quartile and standard deviation for the suitable measures [5]. Additionally, some measures such as convergence and population diversity can be analysed by suitable plots. The next step is the confirmation of the results by hypothesis tests, where the appropriate tests have to be carefully selected [22, 24].

Finally, the results of the performance and behaviour measurements and the consequent analysis are used to answer the stated questions. To this end, they have to be clearly presented and have an obvious relation to the research questions.

4.3 Example: ES and IWO

A first exemplary evaluation aims at showing the feasibility of the approach and demonstrating how much information can be gathered with just a small-scale analysis. We therefore analyse the components of two similar metaheuristics, namely $(\mu + \lambda)$-*Evolution Strategy* (ES) [7] and *Invasive Weed Optimisation* (IWO) [35] in terms of the utilised operators. The similarity results from an identical component structure within the general framework, enabling the evaluation of the incorporated operators in different combinations.

4.3.1 Experiment. The goal is to answer the following questions:

Q1 What are the differences in the performance for the operator combinations?
Q2 Is there a "best" operator (on all test problems)?
Q3 How do the operator combinations influence population diversity?

Table 1: Test functions, all with known optimum 0 at $(0, ..., 0)$.

Name	Function	Range & Dimensions
Sphere	$\sum_{i=1}^{n} x_i^2$	[-5.12, 5.12] 10, 20, 30
Rastrigin	$10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$	[-5.12, 5.12] 10, 20, 30
Ackley	$20 - 20 * \exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + \exp(1)$	[-32, 32] 10, 20, 30

The specific algorithms evaluated are a basic version of the $(\mu + \lambda)$-ES as described in [7] and the IWO [35]. The $(\mu + \lambda)$-ES was chosen as it constitutes a well analysed metaheuristic, while the IWO contains new, yet strongly metaphor-based, operators, which have not been analysed in detail. Both are designed for continuous search spaces and will be evaluated on a small set of continuous real-parameter optimisation problems with different dimensionalities (see Table 1). Though this selection of algorithms is kept small and simple, it will suffice as a proof of concept for the study design. In addition, both algorithms share the same structure, i. e. they contain exactly one component for *initialisation*, *selection*, *generation*, *replacement and update* and *termination* and no *archiving*. Furthermore, the basic versions of both metaheuristics share the same common operators for some components (see Table 2), namely the *initialisation* and *termination*. The operator for *selection* clearly distinguishes both algorithms, while *generation* and *replacement and update* operators are essentially identical but utilise different parameters. We evaluate the original operator sequences of the algorithms as well as different combinations, exchanging operators or utilising parameters from the respective other approach. The combinations are listed in Table 3.

The parameter settings for the operators have to result in a similar performance of the examined algorithms to enable a fair comparison. We therefore conduct a small parameter study to establish the settings for the experiment. However, for now we only evaluate the parameters in the original algorithms. Further, more detailed experiments require finding optimal parameter settings for all individual component combinations. The tested parameter sets can be obtained from Table 4. All combinations of the presented sets are evaluated five times and the best settings are utilised in the respective operators in the further evaluation. Furthermore, the input ranges of the test functions are scaled to $[-1, 1]$, which facilitates the comparison of the *generation* component by a more similar standard deviation σ of the operators. We are aware that comprehensive parameter studies are indispensable for a comprehensive and meaningful analysis. However, for the purpose of this initial example, we select the parameters within these ranges determined by experience and similarity.

Table 2: Components and operators of $(\mu + \lambda)$-ES and IWO.

Component	$(\mu + \lambda)$-ES [7]	IWO [35]
Initialisation	uniform random	uniform random
Selection	uniform random	fitness ranking with linear increase of children
Generation	normally distributed mutation, σ fixed	normally distributed mutation, $\sigma = \frac{(\text{iter}_{\max} - iter)^n}{(\text{iter}_{\max})^n} * (\sigma_{\text{ini}} - \sigma_{\text{end}}) + \sigma_{\text{end}}$
Replacement and Update	keep best from $(\mu + \lambda)$	keep best from parents and children considering $p_{\max}$
Termination	max. number of generations	max. number of generations
Archiving	none	none

Table 3: Operator combinations for experiments (S, G and R for selection, generation, and replacement operators, respectively, utilised by ES and IWO as described in Table 2).

	Combination
ES	$\text{ES}_S + \text{ES}_G + \text{ES}_R$
IWO	$\text{IWO}_S + \text{IWO}_G + \text{IWO}_R$
C1	$\text{ES}_S + \text{ES}_G + \text{IWO}_R$
C2	$\text{IWO}_S + \text{IWO}_G + \text{ES}_R$
C3	$\text{ES}_S + \text{IWO}_G + \text{ES}_R$
C4	$\text{IWO}_S + \text{ES}_G + \text{IWO}_R$
C5	$\text{IWO}_S + \text{ES}_G + \text{ES}_R$
C6	$\text{ES}_S + \text{IWO}_G + \text{IWO}_R$

Table 4: Parameters studied for $(\mu + \lambda)$-ES and IWO.

$(\mu + \lambda)$-ES [7]	IWO [35]
$\mu = \{5, 10, 15, 20, 25\}$	$p_0 = \{5, 10, 15, 20, 25\}$
	$p_{\max} = \{10, 20, 30, 40, 50\}$
$\lambda = \{20, 30, 40, 50, 60\}$	$s_{\min} = \{0, 1, 2, 3, 4\}$
	$s_{\max} = \{2, 3, 4, 5, 6\}$
$\sigma = \{1, 0.8, 0.5, 0.1, 0.01\}$	$\sigma_{\text{ini}} = \{1, 0.8, 0.5, 0.1, 0.01\}$
	$\sigma_{\text{end}} = \{0.1, 0.08, 0.05, 0.01, 0.001\}$
	$n = \{1, 2, 3, 4, 5\}$
gen = 500	gen = 500

All conducted experiments are repeated 50 times. The performance is evaluated by measuring the found optimum. Appropriate statistics for this are mean, standard deviation, median and best and worst value. For the assessment of the algorithmic behaviour, the population diversity in comparison to the development of the best found value is of interest. The changes are plotted against the generations. The population diversity is calculated as described

by Cheng et al. [15]:

$$\mathrm{Div} = \frac{1}{D}\sum_{j=1}^{D}\mathrm{Div}_j \tag{1}$$

with $\mathrm{Div}_j = \frac{1}{m}\sum_{i=1}^{m}|x_{ij} - \overline{x_j}|$.

4.3.2 Results. The source code, written in Rust, and all experimental data, including the hyperparameter study, are available here[1].

The results for the performance of the tested combinations are presented in Table 5. Altogether, the original IWO combination performs best on all functions and dimensionalities, followed by C2 which is very similar in its performance statistics when all test functions are considered. Furthermore, other pairs of combinations with comparable performance for all test problems can be found: ES and C1, C3 and C6, and C4 and C5. All of these have the *selection* and *generation* operators in common. The performance of ES, C1, C4 and C5 is quite similar. They all utilise the *generation* operator of the ES. C3 and C6 utilise ES *selection* but IWO *generation*. They differ from the performance of IWO and C2 especially for the Sphere function, where they show the worst performance of all combinations for 10 and 20 dimensions but perform well for 30 dimensions.

To test the significance of the differences, the Wilcoxon signed-rank test with a significance level of 0.05 was performed on all pairs of evaluated operator combinations on all functions and dimensions. The results are summarised in Table 6.

Figure 2 shows the influence of the different combinations on the population diversity and the respective progress of the found optimum for the iterations of the runs. Here, a similar behaviour can be found for ES, C1, C4 and C5, and IWO, C2, C3 and C6, respectively. These similarities corresponds to the utilisation of the same *generation* method. The combinations with the ES *generation* does not reduce the population diversity as far as those with the IWO *generation* within the 500 iterations of the runs. This also applies to the best function value found within the iterations. Similar results were found for all other functions and dimensions[1] .

4.3.3 Discussion. The results of the experiment show differences in performance and behaviour of the tested operator combinations. Combinations using the *generation* operator of IWO perform better than those utilising the respective operator of the simple ES. Furthermore, the population diversity during the runs of the algorithms also strongly depends on this operator. This is especially interesting as the *generation* operators only differ in the applied standard deviation, which is fixed for the ES and varies with proceeding iterations for IWO. The *selection* operators which differ far more in their design show little influence on performance or behaviour. Only in case of the Sphere function, the performance of combinations with ES *selection* and IWO *generation* differs from those with ES *selection* and ES *generation*. The *replacement and update* component has no discernible influence on performance and population diversity.

With these results, the research questions of this exemplary study can be answered. The differences in the performance of the operator combinations (Q1) are displayed in Table 5. The "best" operator (Q2) found in this small experiment is the *generation* operator of IWO. All combinations utilising this operator clearly show a better performance than those with ES *generation*. In terms of population diversity (Q3), the *generation* operator of IWO again has the most influence. Altogether, though being quite simple in terms of test functions, algorithms and measures, the experiment showed the capabilities and the impact of component and operator studies.

However, some additional aspects need to be considered. The results are limited to the utilised operators, the parameter settings and the test functions. Especially the parameter settings can influence the results strongly. Therefore, a better approach to determine them is necessary. In terms of algorithmic progress, we only evaluated using iterations. It is important to evaluate function evaluations as well, considering that the ES has a predetermined number of function evaluations by a constant number of children, whereas the IWO *selection* can provide different numbers of children in each iteration, depending on the current function values and the parameter settings. Furthermore, other measures are of interest but were not be displayed to keep this study focussed on its immediate goals. Therefore, the results of the example are not comprehensive enough to determine the full capabilities and behavioural influences of the operators.

5 CONCLUSION

Metaheuristic frameworks are formed by combining metaheuristic components. Specific algorithms are then built from operators implementing the component. Starting from previous work we presented a unified approach of a metaheuristic component structure and the order in which they are employed to facilitate the search behaviour (cf. Figure 1). We assume that all metaheuristics can be rebuilt by usage of these components within a general framework without any loss of quality. Building on that assumption, we determine that, by analysing components and the specific operators, they can be attributed with behavioural properties of the metaheuristics where they are typically employed. To perform such analyses we presented a concise general design of experiments.

We then demonstrated the applicability of this design by using it in a small scale study comparing two closely related—algorithmically, but less in the guiding metaphor—metaheuristics, the $(\mu + \lambda)$-Evolution Strategy and the Invasive Weed Optimisation. They use the same component structure but individual operators differ to varying degrees. We then tested the original metaheuristics, as well as versions where we swapped operators between them, on optimising the Sphere, Rastrigin and Ackley functions with 10, 20 and 30 dimensions each. We found that the very similar replacement operator (after some runtime they are identical) had negligible influence on performance. The selection operator which distinguishes the metaheuristics substantially showed only a small influence. The generation operator which is either fixed (in ES) or changes based on runtime (in IWO) was most important for the overall performance and behavioural differences.

In this experiment, operator combinations with the changing generation operator from IWO showed a better performance on all functions compared to the basic generation operator utilised in the ES. Furthermore, it reduces the population diversity more efficiently during runtime. Influences of the selection operator were visible for the Sphere function, where the selection operator of the ES and the

[1] https://git.rz.uni-augsburg.de/luleyleo/mahf-demo

Table 5: Performance of the tested combinations for the test functions and the respective dimensionalities.

$\text{Function}_{\text{Dim}}$	Measures	ES	IWO	C1	C2	C3	C4	C5	C6
Sphere_{10}	mean	2.19e-3	**1.94e-5**	2.32e-3	2.19e-5	2.48e0	1.68e-3	1.79e-3	2.44e0
	std	5.64e-4	**4.86e-6**	5.14e-4	4-95e-6	1.93e0	4.24e-4	4.32e-4	2.23e0
	median	2.15e-3	**1.94e-5**	2.35e-3	2.11e-5	1.98e0	1.72e-3	1.67e-3	1.74e0
	best	6.10e-4	**7.22e-6**	1.17e-3	1.27e-5	7.91e-3	7.75e-4	8.14e-4	2.36e-5
	worst	3.74e-3	**2.76e-5**	3-39e-3	3.41e-5	6.77e0	2-49e-3	2.99e-3	8.89e0
Sphere_{20}	mean	1.57e-2	**1.65e-4**	1.98e-2	9.58e-4	1.41e1	1.29e-2	1.55e-2	1.42e1
	std	2.33e-3	**2.60e-5**	3.02e-3	5.25e-3	5.83e0	1.69e-3	2.77e-3	6.38e0
	median	1.58e-2	**1.72e-5**	1.99e-2	2.15e-4	1.47e1	1.28e-2	1.53e-2	1.46e1
	best	9.84e-3	**9.68e-5**	1.27e-2	1.49e-4	2.42e0	9.49e-3	7.40e-3	4.07e-1
	worst	2.01e-2	**1.06e-4**	2.79e-2	3.73e-2	2.53e1	1.66e-2	2.14e-2	2.68e1
Sphere_{30}	mean	4.12e-2	**4.76e-4**	6.87e-2	5.32e-4	6.98e-4	3.86e-2	4.30e-2	7.08e-4
	std	6.54e-3	5.91e-5	8.40e-3	**5.00e-5**	9.23e-5	4.28e-3	5.64e-3	9.10e-5
	median	4.21e-2	**4.88e-4**	6.95e-2	5.19e-4	7.18e-4	3.88e-2	4.30e-2	7.01e-4
	best	2.33e-2	**3.23e-4**	5.36e-2	4.33e-4	4.60e-4	2.82e-2	2.72e-2	4.68e-4
	worst	5.28e-2	**5.88e-4**	8.94e-2	6.48e-4	9.25e-4	4.75e-2	5.25e-2	8.89e-4
Rastrigin_{10}	mean	2.89e1	**4.07e0**	2.94e1	6.11e0	4.87e0	2.28e1	2.65e1	5.09e0
	std	4.95e0	**1.46e0**	4.16e0	2.95e0	1.88e0	3.46e0	4.33e0	2.06e0
	median	2.96e1	**4.01e0**	2.91e1	5.03e0	5.06e0	2.33e1	2.73e1	5.09e0
	best	1.56e1	**1.02e0**	2.19e1	2.04e0	1.07e0	1.41e1	1.70e1	1.08e0
	worst	3.87e1	**8.00e0**	3.80e1	1.70e1	9.09e0	2.95e1	3.42e1	1.11e1
Rastrigin_{20}	mean	1.13e2	**1.73e1**	1.13e2	3.44e1	2.32e1	9.71e1	1.20e2	2.55e1
	std	1.15e1	**3.95e0**	9.90e0	1.11e1	5.99e0	7.51e0	1.25e1	7.28e0
	median	1.13e2	**1.96e1**	1.14e2	3.39e1	2.39e1	9.87e1	1.20e2	2.59e1
	best	9.14e1	**9.98e0**	8.72e1	1.30e1	1.10e1	7.94e1	8.33e1	1.10e1
	worst	1.38e2	**2.89e1**	1.33e2	6.17e1	3.89e1	1.14e2	1.50e2	4.19e1
Rastrigin_{30}	mean	1.98e2	**3.74e1**	1.99e2	5.30e1	4.82e1	1.98e2	2.06e2	4.82e1
	std	2.56e1	**8.71e0**	1.86e1	1.37e1	1.11e1	2.57e1	2.84e1	1.14e1
	median	1.99e2	**3.55e1**	2.03e2	5.29e1	4.67e1	1.99e2	2.09e2	4.82e1
	best	1.37e2	**2.40e1**	1.21e2	3.10e1	2.72e1	1.62e2	1.46e2	2.64e1
	worst	2.58e2	**6.38e1**	2.62e2	9.97e1	8.69e1	2.61e2	2.91e2	7.60e1
Ackley_{10}	mean	4.73e0	**4.23e-2**	4.99e0	4.72e-2	5.06e-2	4.53e0	4.71e0	5.19e-2
	std	4.52e-1	**5.89e-3**	3.49e-1	8.31e-3	6.42e-3	3.49e-1	5.17e-1	6.25e-3
	median	4.75e0	**4.24e-2**	5.02e0	4.87e-2	5.04e-2	4.51e0	4.79e0	5.30e-2
	best	3.60e0	2.89e-2	3.93e0	**1.82e-2**	3.29e-2	3.45e0	3.46e0	3.17e-2
	worst	5.47e0	**5.39e-2**	5.71e0	6.05e-2	6.27e-2	5.33e0	5.60e0	6.18e-2
Ackley_{20}	mean	7.43e0	**9.52e-2**	7.47e0	1.00e-1	9.96e-2	7.21e0	7.37e0	1.00e-1
	std	3.89e-1	**7.54e-3**	3.54e-1	9.19e-3	8.11e-3	3.71e-1	4.02e-1	8.58e-3
	median	7.49e0	**9.47e-2**	7.50e0	1.00e-1	1.00e-1	7.31e0	7.39e0	1.00e-1
	best	6.19e0	8.19e-2	6.43e0	8.10e-2	**8.06e-2**	6.39e0	6.48e0	8.26e-2
	worst	8.08e0	**1.13e-1**	8.06e0	1.19e-1	1.22e-1	7.84e0	8.18e0	1.19e-1
Ackley_{30}	mean	8.82e0	**1.29e-1**	9.48e0	1.34e-1	1.69e-1	8.65e0	8.71e0	1.67e-1
	std	3.54e-1	**9.19e-3**	2.40e-1	1.30e-2	1.34e-2	3.01e-1	3.45e-1	1.41e-2
	median	8.83e0	**1.29e-1**	9.50e0	1.36e-1	1.71e-1	8.65e0	8.77e0	1.70e-1
	best	8.00e0	9.52e-2	9.01e0	**9.16e-2**	1.38e-1	7.55e0	8.01e0	1.18e-1
	worst	9.50e0	**1.44e-1**	9.97e0	1.54e-1	1.94e-1	9.18e0	9.27e0	1.94e-1

generation operator of IWO resulted in worse performance for the dimensions 10 and 20, but a good performance for 30 dimensions. This answered the three specific research questions stated for our exemplary study. However, it only emphasises the need for further studies as the results are limited to the small range of problems, operator combinations and measures utilised for these experiments.

Component/operator studies—as presented here—help predicting the behaviour of metaheuristics based on which operators they use by comparing them to metaheuristics using similar operators. We furthermore assume that these studies will provide a source of inspiration for detailed theoretical studies, as they point out interesting behavioural features that are worth investigating theoreti-

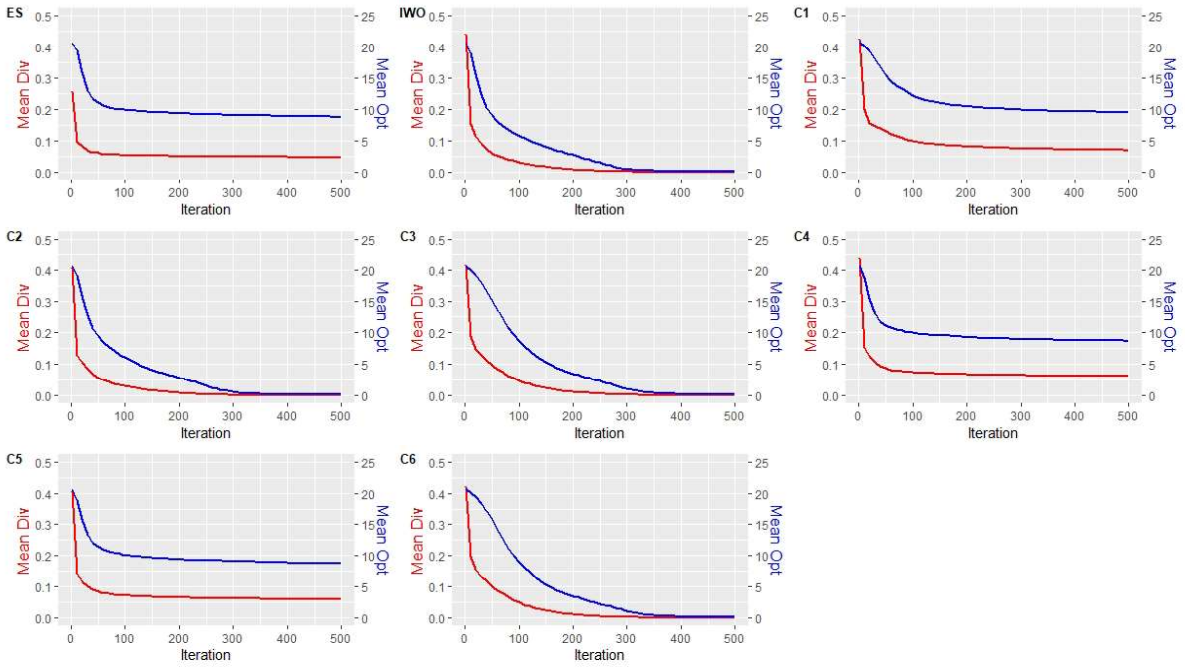

Figure 2: Progression of mean population diversity (Mean Div) and mean found optimum (Mean Opt) during the iterations of the runs for all combinations on the 30 dimensional Ackley function.

Table 6: Results of Wilcoxon signed-rank test. For each function (Sphere (S), Rastrigin (R), Ackley (A)) and algorithm, we show all other algorithms *without* significant difference.

	ES	IWO	C1	C2	C3	C4	C5	C6
S_{10}	C1		ES		C6	C5	C4	C3
S_{20}	C5				C6		ES	C3
S_{30}	C5				C6		ES	C3
R_{10}	C1		ES	C6	C6			C2, C3
R_{20}	C1		ES		C6			C3
R_{30}	C1, C4, C5		ES	C3, C6	C2, C6	ES, C5	ES, C4	C2, C3
A_{10}	C5				C6		ES	C3
A_{20}	C1, C5		ES, C5	C3, C6	C2, C6	C5	ES, C1, C4	C2, C3
A_{30}	C5	C2		IWO	C6	C5	C4, ES	C3

cally. Altogether, component/operator studies are highly suitable to deepen the understanding of metaheuristics.

6 FUTURE WORK

For this approach to be ultimately successful, it is necessary to derive components and operators from many more metaheuristics and incorporate them into the analysis. This will be the immediate next step. However, it has to be considered that some metaheuristic components might not fit into the current concept for a unified framework, so it is possible that some refinement is necessary. Furthermore, a comprehensive set of performance and behavioural measures has to be established, building upon the first descriptions in Section 4.2. In terms of analysis, it is important to formulate adequate hypotheses that correspond to the research questions. Only then is it possible to start large-scale studies to answer these questions. These studies will start with analysing known metaheuristic approaches and their established operator settings to verify the results by comparing them to existing studies. Then, not yet utilised operators and additional component combinations will be evaluated. Additionally, at this point, it is important to include the operator related hyperparameters in the evaluation. Without hyperparameter studies, the results of the component evaluations would not adequately represent the actual influences.

Another problem that remains to be solved is the presentation of the results and gathered data. Due to the extensiveness of the approach this is an important issue. All information has to be made publicly available. However, an appropriate clear and descriptive format has to be found and existing tools, e. g. the IOHanalyzer of the IOHprofiler platform [21], have to be evaluated to determine their suitability for behavioural evaluation.

REFERENCES

[1] Otman Abdoun and Jaafar Abouchabaka. 2012. A Comparative Study of Adaptive Crossover Operators for Genetic Algorithms to Resolve the Traveling Salesman Problem. *International Journal of Computer Applications (0975 - 8887) Volume 31 - No.11, October 2011* (March 2012). arXiv:cs.NE/1203.3097

[2] Thomas Bäck. 1994. Selective pressure in evolutionary algorithms: a characterization of selection mechanisms. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*. IEEE. https://doi.org/10.1109/icec.1994.350042

[3] Sunith Bandaru and Kalyanmoy Deb. 2016. Metaheuristic Techniques. In *Decision Sciences*. CRC Press, 693–750. https://doi.org/10.1201/9781315183176-12

[4] Richard S. Barr, Bruce L. Golden, James P. Kelly, Mauricio G. C. Resende, and William R. Stewart. 1995. Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics* 1, 1 (1995), 9–32. https://doi.org/10.1007/bf02430363

[5] Thomas Bartz-Beielstein, Carola Doerr, Daan van den Berg, Jakob Bossek, Sowmya Chandrasekaran, Tome Eftimov, Andreas Fischbach, Pascal Kerschke, William La Cava, Manuel Lopez-Ibanez, Katherine M. Malan, Jason H. Moore, Boris Naujoks, Patryk Orzechowski, Vanessa Volz, Markus Wagner, and Thomas Weise. 2020. Benchmarking in Optimization: Best Practice and Open Issues. (2020). arXiv:cs.NE/2007.03488

[6] Florentin Batrinu, Enrico Carpaneto, and Gianfranco Chicco. 2005. A unified scheme for testing alternative techniques for distribution system minimum loss reconfiguration. In *2005 International Conference on Future Power Systems*. IEEE. https://doi.org/10.1109/fps.2005.204275

[7] Hans-Georg Beyer and Hans-Paul Schwefel. 2002. Evolution strategies - A comprehensive introduction. *Natural Computing* 1, 1 (2002), 3–52. https://doi.org/10.1023/a:1015059928466

[8] Dinabandhu Bhandari, Nikhil R. Pal, and Sankar K. Pal. 1994. Directed mutation in genetic algorithms. *Information Sciences* 79, 3-4 (1994), 251–270. https://doi.org/10.1016/0020-0255(94)90123-6

[9] Mauro Birattari, Luis Paquete, Thomas Stützle, and Klaus Varrentrapp. 2001. *Classification of Metaheuristics and Design of Experiments for the Analysis of Components*. Technical Report.

[10] Tobias Blickle and Lothar Thiele. 1996. A Comparison of Selection Schemes Used in Evolutionary Algorithms. *Evolutionary Computation* 4, 4 (1996), 361–394. https://doi.org/10.1162/evco.1996.4.4.361

[11] Christian Blum and Andrea Roli. 2003. Metaheuristics in combinatorial optimization. *Comput. Surveys* 35, 3 (2003), 268–308. https://doi.org/10.1145/937503.937505

[12] Stephan Blum, Romanas Puisa, Jörg Riedel, and Marc Wintermantel. 2005. Adaptive Mutation Strategies for Evolutionary Algorithms. In *The Annual Conference: EVEN at Weimarer Optimierungsund Stochastiktage*, Vol. 2.

[13] Rainer E. Burkard, Eranda Çela, Stefan E. Karisch, and Franz Rendl. 2011. QAPLIB. (2011). https://coral.ise.lehigh.edu/data-sets/qaplib/

[14] Marco Caserta and Stefan Voß. 2009. Metaheuristics: Intelligent Problem Solving. In *Matheuristics*. Springer US, 1–38. https://doi.org/10.1007/978-1-4419-1306-7_1

[15] Shi Cheng, Yuhui Shi, Quande Qin, Qingyu Zhang, and Ruibin Bai. 2014. Population Diversity Maintenance In Brain Storm Optimization Algorithm. *Journal of Artificial Intelligence and Soft Computing Research* 4, 2 (2014), 83–97. https://doi.org/10.1515/jaiscr-2015-0001

[16] Gianfranco Chicco and Andrea Mazza. 2020. Metaheuristic Optimization of Power and Energy Systems: Underlying Principles and Main Issues of the 'Rush to Heuristics'. *Energies* 13 (2020), 5097. https://doi.org/10.3390/en13195097

[17] Matej Črepinšek, Shih-Hsi Liu, and Marjan Mernik. 2013. Exploration and exploitation in evolutionary algorithms. *Comput. Surveys* 45, 3 (2013), 1–33. https://doi.org/10.1145/2480741.2480752

[18] Jorge M. Cruz-Duarte, José C. Ortiz-Bayliss, Iván Amaya, Yong Shi, Hugo Terashima-Marín, and Nelishia Pillay. 2020. Towards a Generalised Metaheuristic Model for Continuous Optimisation Problems. *Mathematics* 8, 11 (2020), 2046. https://doi.org/10.3390/math8112046

[19] Jesica de Armas, Eduardo Lalla-Ruiz, Surafel Luleseged Tilahun, and Stefan Voß. 2021. Similarity in metaheuristics: a gentle step towards a comparison methodology. *Natural Computing* (2021). https://doi.org/10.1007/s11047-020-09837-9

[20] Kalyanmoy Deb and Nikhil Padhye. 2010. Development of efficient particle swarm optimizers by using concepts from evolutionary algorithms. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation - GECCO '10*. ACM Press. https://doi.org/10.1145/1830483.1830492

[21] Carola Doerr, Hao Wang, Furong Ye, Sander van Rijn, and Thomas Bäck. 2018. IOHprofiler: A Benchmarking and Profiling Tool for Iterative Optimization Heuristics. *arXiv e-prints:1810.05281* (2018). arXiv:1810.05281 https://arxiv.org/abs/1810.05281

[22] Tome Eftimov, Gašper Petelin, and Peter Korošec. 2020. DSCTool: A web-service-based framework for statistical comparison of stochastic optimization algorithms. *Applied Soft Computing* 87 (2020), 105977. https://doi.org/10.1016/j.asoc.2019.105977

[23] Evert Haasdijk and Jacqueline Heinerman. 2018. Quantifying Selection Pressure. *Evolutionary Computation* 26, 2 (2018), 213–235. https://doi.org/10.1162/evco_a_00207

[24] A. Hanif Halim, I. Ismail, and Swagatam Das. 2020. Performance assessment of the metaheuristic optimization algorithms: an exhaustive review. *Artificial Intelligence Review* 54, 3 (2020), 2323–2409. https://doi.org/10.1007/s10462-020-09906-6

[25] Nikolaus Hansen, Anne Auger, Raymond Ros, Olaf Mersmann, Tea Tušar, and Dimo Brockhoff. 2020. COCO: A Platform for Comparing Continuous Optimizers in a Black-Box Setting. *Optimization Methods and Software* (2020). https://doi.org/10.1080/10556788.2020.1808977

[26] Ahmad Hassanat, Khalid Almohammadi, Esra'a Alkafaween, Eman Abunawas, Awni Hammouri, and V. B. Surya Prasath. 2019. Choosing Mutation and Crossover Ratios for Genetic Algorithms—A Review with a New Dynamic Approach. *Information* 10, 12 (2019), 390. https://doi.org/10.3390/info10120390

[27] John N. Hooker. 1994. Needed: An Empirical Science of Algorithms. *Operations Research* 42, 2 (1994), 201–212. https://doi.org/10.1287/opre.42.2.201

[28] John N. Hooker. 1995. Testing heuristics: We have it all wrong. *Journal of Heuristics* 1, 1 (1995), 33–42. https://doi.org/10.1007/bf02430364

[29] Khalid Jebari and Mohammed Madiafi. 2013. Selection Methods for Genetic Algorithms. *International Journal of Emerging Sciences* 3 (2013), 333–344.

[30] Krzysztof Krawiec, Christopher Simons, Jerry Swan, and John R. Woodward. 2018. *Metaheuristic Design Patterns: New Perspectives for Larger-Scale Search Architectures*. IGI Global, 1–36.

[31] Antonio LaTorre, Daniel Molina, Eneko Osaba, Javier Del Ser, and Francisco Herrera. 2020. Fairness in Bio-inspired Optimization Research: A Prescription of Methodological Guidelines for Comparing Meta-heuristics. (2020). arXiv:cs.NE/2004.09969

[32] Siew Mooi Lim, Abu Bakar Md Sultan, Md Nasir Sulaiman, Aida Mustapha, and K. Y. Leong. 2017. Crossover and Mutation Operators of Genetic Algorithms. *International Journal of Machine Learning and Computing* 7 (2017), 9–12. https://doi.org/10.18178/ijmlc.2017.7.1.611

[33] Michael A. Lones. 2014. Metaheuristics in nature-inspired algorithms. In *Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion - GECCO Comp '14*. ACM Press. https://doi.org/10.1145/2598394.2609841

[34] Michael Adam Lones. 2019. Mitigating Metaphors: A Comprehensible Guide to Recent Nature-Inspired Algorithms. *SN Computer Science* 1, 49 (2019). https://doi.org/10.1007/s42979-019-0050-8 arXiv:cs.NE/http://arxiv.org/abs/1902.08001v1

[35] A. R. Mehrabian and C. Lucas. 2006. A novel numerical optimization algorithm inspired from weed colonization. *Ecological Informatics* 1 (2006), 355–366. https://doi.org/10.1016/j.ecoinf.2006.07.003

[36] Daniel Molina, Javier Poyatos, Javier Del Ser, Salvador García, Amir Hussain, and Francisco Herrera. 2020. Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration versus Algorithmic Behavior, Critical Analysis and Recommendations. *Cognitive Computation* (2020). https://doi.org/10.1007/s12559-020-09730-8

[37] Eman Mostafa, Mohamed Abdel-Nasser, and Karar Mahmoud. 2017. Performance evaluation of metaheuristic optimization methods with mutation operators for combined economic and emission dispatch. In *2017 Nineteenth International Middle East Power Systems Conference (MEPCON)*. IEEE. https://doi.org/10.1109/mepcon.2017.8301304

[38] E. Osaba, R. Carballedo, F. Diaz, E. Onieva, I. de la Iglesia, and A. Perallos. 2014. Crossover versus Mutation: A Comparative Analysis of the Evolutionary Strategy of Genetic Algorithms Applied to Combinatorial Optimization Problems. *The Scientific World Journal* 2014 (2014), 1–22. https://doi.org/10.1155/2014/154676

[39] Nikhil Padhye, Piyush Bhardawaj, and Kalyanmoy Deb. 2012. Improving differential evolution through a unified approach. *Journal of Global Optimization* 55, 4 (2012), 771–799. https://doi.org/10.1007/s10898-012-9897-0

[40] Suvarna Patil and Manisha Bhende. 2014. Comparison and Analysis of Different Mutation Strategies to improve the Performance of Genetic Algorithm. In *(IJCSIT) International Journal of Computer Science and Information Technologies*, Vol. 5.

[41] Pupong Pongcharoen, Watthanapol Chainate, and Chaowanee Samranpun. 2007. Exploration of Genetic Parameters and Operators through Travelling Salesman Problem. *ScienceAsia* 33, 2 (2007), 215. https://doi.org/10.2306/scienceasia1513-1874.2007.33.215

[42] Gerhard Reinelt. 2013. TSPLIB. (2013). http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

[43] Nisha Saini. 2017. Review of Selection Methods in Genetic Algorithms. *International Journal of Engineering and Computer Science* 6, 12 (2017), 22261–22263. http://ijecs.in/index.php/ijecs/article/view/2562

[44] Andreas Scheibenpflug and Stefan Wagner. 2013. An Analysis of the Intensification and Diversification Behavior of Different Operators for Genetic Algorithms. In *Computer Aided Systems Theory - EUROCAST 2013*. Springer Berlin Heidelberg, 364–371. https://doi.org/10.1007/978-3-642-53856-8_46

[45] Andreas Scheibenpflug, Stefan Wagner, Erik Pitzer, Bogdan Burlacu, and Michael Affenzeller. 2012. On the analysis, classification and prediction of metaheuristic algorithm behavior for combinatorial optimization problems. *24th European Modeling and Simulation Symposium, EMSS 2012* (2012), 368–372.

[46] Anupriya Shukla, Hari Mohan Pandey, and Deepti Mehrotra. 2015. Comparative review of selection techniques in genetic algorithm. In *International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*. https://doi.org/10.1109/ablaze.2015.7154916

[47] Qun Song and Simon Fong. 2016. Brick-Up Metaheuristic Algorithms. In *5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)*. https://doi.org/10.1109/iiai-aai.2016.100

[48] N. Soni and T. Kumar. 2014. Study of Various Crossover Operators in Genetic Algorithms. In *(IJCSIT) International Journal of Computer Science and Information Technologies*, Vol. 5.

[49] N. Soni and T. Kumar. 2014. Study of Various Mutation Operators in Genetic Algorithms. In *(IJCSIT) International Journal of Computer Science and Information Technologies*, Vol. 5.

[50] Helena Stegherr, Michael Heider, and Jörg Hähner. 2020. Classifying Metaheuristics: Towards a unified multi-level classification system. *Natural Computing* (2020). https://doi.org/10.1007/s11047-020-09824-0

[51] Jörg Stork, A. E. Eiben, and Thomas Bartz-Beielstein. 2020. A new taxonomy of global optimization algorithms. *Natural Computing* (2020). https://doi.org/10.1007/s11047-020-09820-4

[52] A. J. Umbarkar and P. D. Sheth. 2015. CROSSOVER OPERATORS IN GENETIC ALGORITHMS: A REVIEW. *ICTACT Journal on Soft Computing* 06 (2015), 1083–1092. https://doi.org/10.21917/ijsc.2015.0150

[53] Xin-She Yang. 2011. Metaheuristic Optimization: Algorithm Analysis and Open Problems. In *Experimental Algorithms*. Springer Berlin Heidelberg, 21–32. https://doi.org/10.1007/978-3-642-20662-7_2

[54] Furong Ye, Hao Wang, Carola Doerr, and Thomas Bäck. 2020. Benchmarking a $(\mu + \lambda)$ Genetic Algorithm with Configurable Crossover Probability. In *Parallel Problem Solving from Nature – PPSN XVI*, Thomas Bäck, Mike Preuss, André Deutz, Hao Wang, Carola Doerr, Michael Emmerich, and Heike Trautmann (Eds.). Springer International Publishing, Cham, 699–713.